2023011932 曲浩瑜

一、
1. 管理计算机硬件与软件资源的软件集合
2. 及时响应外部事件并处理的操作系统
3. 资源可提供给多个进程，但一段时间只有一个进程可使用

二、
1. 硬件 软件
2. 硬件资源 软件资源
3. 异步性 共享性

一、
1. 访问共享资源的代码段
2. 多个进步完成共同任务 协调配合的机制

二、
1. 动态 独立 异步性
2. 动态 静态
3. 间接制约
4. 共享存储 消息传递 管道
5. 就绪 运行 阻塞

三、大题
1. S_1无直接前驱

$S_1 \to S_2$
$S_1 \to S_3$
$S_2 \to S_4$
$S_2 \to S_5$
$S_3 \to S_6$
$S_3, S_4, S_5 \to S_7$

$sem\_S_2 = 0$ 控制S_2等S_1
$sem\_S_3 = 0$ 控制S_3等S_1
$sem\_S_4 = 0$ 控制S_4
$sem\_S_5 = 0$ 控制S_5
$sem\_S_6 = 0$ 控制S_6
sem_S_{71}, sem_S_{72}, sem_S_{73} 用于控制S_7

```
sem  mutex = 1
sem  empty = 5
sem  apple = 0
sem  orange = 0

void father(){
    while(1){
        int fruit = rand()%2
        P(empty)
        P(mutex)
        if(fruit==0){
            V(apple);}
        else {
            V(orange);
        }
        V(mutex)
    }
}

void son(){
    while(1){
        P(orange)
        P(mutex)
        V(empty)
        V(mutex)
    }
}

void daughter(){
    while(1){
        P(apple)
        P(mutex)
        V(empty)
        V(mutex)
    }
}
```

3. PA → [1] → PB → [2] → PC

```
sem  empty1 = 1
sem  full1 = 0
sem  empty2 = 1
sem  full2 = 0

sem  empty3 = 1
sem  full3 = 0
sem  mutex1 = 1
sem  mutex2 = 1

void PA(){
    while(1){
        P(empty1)
        P(mutex1)
        print("放入1")
        V(mutex1)
        V(full1)
    }
}

void PB(){
    while(1){
        P(full1)
        P(mutex1)
        V(mutex1)
        V(empty1)
        P(empty2)
        P(mutex2)
        print(放入2)
        V(mutex2)
        V(full2)
    }
}

void PC(){
    while(1){
        P(full2)
        P(mutex2)
        print(2)
        V(mutex2)
        V(empty2)
    }
}
```

4.

```
int count=0
sem mutex=1   empty=1   full=0
sem   payment=0     receipt=0

void guest(){
    P(mutex)
    if(count>=N)
        signal(mutex)
    else{
        count++
        V(mutex)
        P(empty)
        P(mutex)
        count--;
        V(mutex)
        V(full)
        理发
        付费
        V(payment)
        P(receipt)
        V(empty)
    }
}

void Barber(){
    while(1){
        P(full)
        P(payment)
        V(receipt)
    }
}
```

一、

1. 按算法对进程分配处理机

2. 从作业提高到系统到作业完成的时间间隔

3. 多进程竞争资源导致循环等待的状态

二、

1. 提交 后备 完成

2. 死锁预防 死锁避免 死锁解除 死锁检测

三、大题

1.

先来先服务： A→B→C→D→E

周转时间 3 7 9 12 12 $\bar{t}=8.6$

带权周转时间： 1 $\frac{7}{6}$ $\frac{9}{4}$ $\frac{12}{5}$ 6 $\bar{T}=2.56$

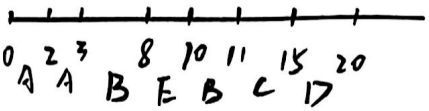
SJF A→B→E→C→D

周： 3 7 3 11 14 $\bar{t}=7.6$

权周： 1 $\frac{7}{6}$ 1.5 2.75 2.8 $\bar{T}=1.84$

~~HRRN~~

抢占式



周： $\bar{t}=(3+9+2+11+14)\div5=7.8$

权周： $\bar{T}=(1+1.5+1+2.75+2.8)\div5=1.81$

HRRN

周： $\bar{t}=(3+7+9+7+14)\div5=8$

权周： $\bar{T}=(1+1.17+2.25+3.5+2.8)\div5=2.14$

RR 周： $\bar{t}=(5+6+12+9+13+6)\div5=10.4$

权周 $\bar{T}=(1.67+2.67+3+2.6+3)\div5=2.59$

2.

检测点A = 20 - 0 - 10 = 10ms

B = 50 - 0 - 10 = 40ms

C = 50 - 10 - 15 = 35 ms

A最小 先执行A

10ms后

B = 50 - 10 - 10 = 30 ms

C = 50 - 10 - 15 = 25 ms

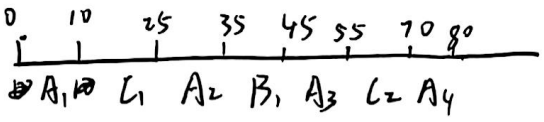
C最小，先执行C

25ms后

A = 20 - 25 - 10 = -15ms

B = 50 - 25 - 10 = 15ms

先执行A2



3.

Allocation = Max - Need

Pro	All
P_0	0 0 3 2
P_1	1 0 0 0
P_2	1 3 5 4
P_3	0 3 3 2
P_4	0 0 1 4

	work	Need	All	work+All	Finish
	A B C D	A B C D	A B C D	A B C D	
P_0	1 6 2 2	0 0 1 2	0 0 3 2	1 6 5 4	T
P_3	1 6 5 4	0 6 5 2	0 3 3 2	1 9 8 6	T
P_4	1 9 8 6	0 6 5 6	0 0 1 4	1 9 9 10	T
P_1	1 9 9 10	1 7 5 0	1 0 0 0	2 9 9 10	T
P_2	2 9 9 10	2 3 5 6	1 3 5 4	3 12 14 14	T

安全序列为 {P_0, P_3, P_4, P_1, P_2}

(2) ① Request(1,2,2,2) ≤ Need(2,3,5,6)

② Request ≤ Avai(1,6,2,2)

③假定分配资源。

Ava = (0,4,0,0)

ALL = (2,5,7,6)

Need = (1,1,3,4)

④ Need$_i$ ≤ (0,4,0,0) 不成立

故进入不安全状态

(3) 并不是

因为进程没有申请新的资源，只有进程提出新的请求，导致循环等待，进入死锁。